

On the integration of biomedical knowledge bases: problems and solutions

Marco Fato^{2§}, Enrico Giunchiglia^{1§}, Ivan Porro^{2§}, Luca Vassalli^{1§}

¹ Systems and Technologies for Automated Reasoning laboratory, DIST, University of Genova, Viale Causa 13, 16145 Genova, Italy

² Bioengineering and Bioimages laboratory (Biolab), DIST, University of Genova, Viale Causa 13, 16145 Genova, Italy

[§]Corresponding author

Email addresses:

EG: giunchiglia@unige.it

LV: lucanl@star.dist.unige.it

MF: fantomas@dist.unige.it

IP: pivan@dist.unige.it

Abstract

Background

The problem of integrating different biomedical knowledge bases has to face lots of problems than are even more difficult in the biomedicine field. We moved from the need of the integration of KEGG and Reactome information in particular about the metabolic pathways. Since current systems like SRS, Kleisli and P/FDM cannot be used to integrate this particular information, we developed a custom solution.

Results

We considered the pros and cons of the two main modeling techniques for such a systems: the GAV and LAV approaches. We chose one of them according to the properties these approaches have and the particular requirements we had to meet. We chose also one among the possible ways to access data to improve reliability, soundness and completeness of the system.

Conclusions

Our system can syntactically integrate the information described in its model for the two data sources Reactome and KEGG in a modular, efficient, effective and transparent way. It is going to be used inside an already existing web service as a service to service but can also be used as a standalone web service.

Background

We witness a rapid increase in the number of structured information sources that are available on Internet. Making a simple search may involve visiting more websites and merging the retrieved results or filling a form with all the results of a precedent search. Carrying out these operations manually is not only error prone but also highly time consuming for the time needed to access the different data sources but also for the training phase that their interfaces require to be profitably used. This is not

desirable: an user should be able to focus on *what* he is looking for rather than thinking about *how* to obtain the answer.

Addressing this problem requires an uniform interface to this multitude of information sources provided by a system able to create a query execution plan including all the relevant data sources. This system has also to be able to refine the information returned by the individual information sources, for instance identifying when the same entity is named in various ways in order to underline mismatches and hide overlaps.

However this arises lots of different issues. Internet is not a stable environment, new data sources become available and others change their structure. Different exchange formats or different versions of a same format are used by different organizations. The internal representation of the data source is often unknown. Sometimes the only provided access to the source is your own browser adding the issue of extracting the information from the semi-structured HTML page [1].

Even though the heterogeneity and complexity of the problems involved, systems as Information Manifold [2] and Ariadne [3] are capable of handling them properly. However biomedicine is a field where some extra care has to be used. In fact these systems are based on the assumption that the same entity can be unambiguously identified among the data sources. This is a misleading assumption in biomedicine since for instance the same protein may have different names in different systems. Diverse research groups often create their own knowledge base according to their own needs: the same information can be presented from various points of view making tricky to identify it as the same entity in an automated way. An attempt to address these issues is behind the fact that this domain has embraced web standards, such as web services and XML formats, more than other domains. Nevertheless those solutions often hide further troubles: bugs or poor documentation, lots of different XML formats or different version of the same format [4], lack of tools to handle novel formats. Ontologies matching research has been performed to face some of these problems but it seems to not have reach a very advanced and settled state yet [5].

In spite of all the difficulties some interesting bioinformatics integration systems were developed in the last few years. Every one addresses some particular needs and problems. SRS [6] is a fast and effective data retrieval system for indexed flat-file text data sources. Kleisli [7] integrates heterogeneous data sources using a query language named CPL, although it is not very transparent from the underlying structure. P/FDM [8] is a research prototype that provides distributed data sources access through CORBA which does not use wrappers. Knoblock et al. work is a very flexible and easily extensible system, but so far it integrates a narrow set of information of the selected data sources [9]. Other systems focus more on the semantic knowledge representation issue using ontologies that allow sophisticated reasoning about biological concepts rather than a syntactic integration [10, 11]. These integration systems can be extended only to systems which provide their data as XML files.

Despite all these efforts there are still lots of different knowledge bases which are not integrated yet. Therefore new integration systems or an extension of the existing ones is necessary. The key point here is that the new integration systems have to be exposed in manner that they can become in turn the input to new integration systems. This composition will permit to cover more and more informations in disparate fields

of research.

Our research goal was the integration of two well known and established knowledge bases: KEGG [12] and Reactome [13]. In particular we moved from the need of complete and reliable information about the metabolic pathways in humans. The implemented system has the property to be modular so to be easily extended to other databases. With extension, we mean both an enrichment of the information extracted from the data sources already integrated or an increase in their number. Our system can be exposed both as an end user web service or as a service for a service in a composition of web services, using for instance a BPEL orchestration paradigm. Moreover, the underlying databases are completely transparent to the end user. The integrated information are exposed as database relations so that the end user just need to express a query over this global schema and the system will provide the answer. In this way multiple different interfaces can be created according to the kind of information that the front end writer wants to retrieve, keeping simple the single specific interface. In addition, in a service to service viewpoint, the developer of the web service is guaranteed that the system will query the right databases and return the needed information, as long as he/she adheres to the model of the integrated information.

Results and discussion

GAV and LAV approaches to mediated schema design

One of the fundamental quality an integration system is supposed to have is to be completely transparent to the end user. Who makes the queries over the mediated schema cannot be bothered with tedious details connected to the data sources that have to be interrogated. It is due to the system to understand which sources are relevant and which is the best query plan. This allows also the benefit that who develops the front end interface can work in total independence from who deploys the core application or the wrappers.

These are the reasons for which a modeling of the system is needed. The model must present the information that the system integrates. The optimality would be achieved if this model represented an union of all the information that can be extracted from the separated knowledge bases.

Using such a model implies that the end user will query the system using the language of the mediated schema. These queries have to be translated to the corresponding data source native language. The translation, or rewriting, is carried out using a set of rules which describe the relation between the mediated schema and the local data sources. In principle, it is possible describing the mediated schema in any language, even first order logic. On the other hand for every language you should consider the possibility to generate a sound and complete rewriting [15]. Several approaches have been explored in which restricted forms of first-order formulas have been used in source descriptions, and effective accompanying reformulation algorithms have been presented. Two of the most effective approaches are: global as view (GAV) [16] and local as view (LAV) [17].

In the GAV approach the mediated schema is described in terms of the data source relations. So for each relation R in the mediated schema, we write a query over the

source relations specifying how to obtain R's tuples from the sources.

We can imagine for example to have the following data sources:

- ◆ DB1 (Pathway_Name, Pathway_ID, Description, Molecule)
- ◆ DB2 (Pathway_ID, Organism)

so that DB1 has a relation that lists pathway names, their ids, their short descriptions and the molecules that they contain while DB2 lists pathway ids and the correspondent organism. In this case the GAV description of a mediated schema made up only by these two databases using a Datalog notation would be:

- ◆ Pathway(Pathway_Name, Description, Organism) :-
DB1 (Pathway_Name, Pathway_ID, Description, Molecule),
DB2 (Pathway_ID, Organism)
- ◆ Connection_Molecule (Pathway_Name, Molecule) :-
DB1 (Pathway_Name, Pathway_ID, Description, Molecule)

Please note that if the relation “Pathway” was described in this way it would mean that only elements whose id is present in both DB1 and DB2 should be returned to the user.

The main advantage of the GAV approach is that a sound and complete rewriting in terms of data sources relation can be done in polynomial time, eliminating the need of a time consuming containment checking, necessary in all the other strategies. Another benefit is that this approach can be easily lead to a hierarchy of mediated schemas. On the other hand adding a new source is not a trivial task since it requires to figure out all the possible ways in which it can be used to obtain tuples for each of the relations in the mediated schema. For this reason the ability of this strategy to scale up to a great number of different data sources is limited.

On the contrary in the LAV method the data sources are expressed in terms of the mediated schema. For instance we can suppose that we have two sources, where the first one contain human pathway names and their descriptions while a second one human pathway names and proteins. The correspondent rules are:

- ◆ DB1 (PName, Description) :- Pathway (PName, Description, Organism, PID),
Organism= “homo sapiens”
- ◆ DB2(PName, MolecName) :- Molecule (PID, MolecName, Class),
Pathway (PName, Description, Organism, PID),
Class = “protein”, Organism= “homo sapiens”

In this case we can notice that rewriting the user query over the mediated schema is much more difficult. For example if we wanted a query which asks for all the molecule and pathway names in both humans and cows, we would use the query:

q(PName, MolecName) :- Molecule (PID, MolecName, Class),
Pathway (PName, Description, Organism, PID),
(Organism=“homo sapiens” or Organism=“bos taurus”)

The reformulated query on the sources would be:

q'(PName, MolecName) :- DB1(PName, Description), DB2(PName, MolecName)

This query would not answer the user's query with all the information requested but only with the maximum amount of information which is possible to obtain using just DB1 and DB2. For instance is not possible to get any molecule which is in the “bos taurus” pathway, since such a pathway is not present in the two integrated databases.

We would say that the rewriting above is a maximally-contained rewriting but not an equivalent one [15].

The LAV approach is more flexible than the GAV one since every data source is described in isolation. Adding new sources requires a little effort, and constraints on the data available to a data source are easily specified since they can just be added to the source description. On the other hand the rewriting of the rules requires expensive containment checking to verify that it is sound and complete. More precisely if the query does not contain comparison predicates and is a conjunction of predicates, the problem of finding a contained rewriting of a query using a set of views is NP-complete (requiring exponential time to be solved) [18] otherwise it may be even intractable (an algorithm to solve the problem cannot be found) [19].

The approach we have chosen for our system is a GAV one. This approach is the best solution to integrate a relative small number of well established sources that are less likely to frequently change, like the two selected. Our system checks for duplicated entries and filters them from the output, extracts a considerable amount of information from the integrated data sources, may combine in any way the integrated sources in a flow of calls to different sources. If it is true that it is more complicated to extend the system than using a LAV approach, on the other hand the system can be extended with a reasonable amount of effort to new sources from which are extracted the same kind of information that are extracted from the current ones. For instance to extend the example we saw before to the new source DB3 (Pathway_ID, Organism) would require just to add this source in the first rule. That model of information retrieved from new sources might be equal to the current ones is reasonable since we believe that in the biomedical field the integrator systems are mainly aimed toward some specific issues. In our case the project moved from the need for the integration of metabolic pathways, and when new sources will be added in the future they will likely be quite similar to the ones already present in the system, not requiring large change to the overall structure.

Services to users or service to services

The response time of the system has been improved with the fastest Java data structures and with the use of a GAV modeling approach. Also the algorithms for the retrieving of the information from the data sources and for their merging have been shrewdly optimized. Nevertheless the response time of the system can vary from mere decades of seconds for queries which involve both data sources with data already cached in Reactome, to some dozens of minutes for some particular complicated tasks which require querying more times Reactome with not cached data. For this reason, even if the system can be presented as a stand alone integration system with a public interface accessible by a browser, to be able to extract all the available information, we are now focusing to use it inside an already developed application [14] as a service to a service.

Since it exposes the integrated information as a data source itself, it may be easily integrated by other integration systems. In fact we believe that a response to the increase in the number of data sources may be a proliferation in the integration systems, every one with its specific task and its main target sources. The composition of these web services will allow more and more precise and complete biomedical searches.

Web services versus XML files

In our system the information that it is possible to retrieve from the data sources is heterogeneous. Indeed it is possible to get information about pathways, molecules and reactions. For this reason each of the two data sources was modeled as a group of relations instead of a single relation as in other approaches [9]. Each relation is considered in isolation, thus if the knowledge bases somehow changed, we would need just to change a part of the wrapper instead of the whole of it.

There are lots of different ways to extract at least part of the information of the two knowledge bases. Reactome is available in SBML format level 2 version 1, in BioPax format level 2 and as a SOAP based Web Service API. KEGG is available as Web Service API, in KGML format, in SBML format level 1 version 2, in BioPax level 1.

SBML [20] is a well established XML format in systems biology which can describe also molecular pathways. Unluckily it cannot be used to integrate the information that our system handles, since only a limited subset of Reactome is available in this format. KEGG is available in SBML using a free conversion tool. Unfortunately this tool is dated, so that in order to work properly requires the version 0.4 of the KGML files. Unfortunately the current version is the 0.6.1. That means that the conversion of all the data released between the 0.4 and the 0.6 releases would not be based on the indication of these KGML files and so would be potentially less correct. We thought it would be difficult that an end user would ever use an integrator system which retrieves potentially corrupted and lossy information.

BioPax [21] is a younger format which supports the Web Ontology Language specifications, thus allowing reasoning on the hierarchical structure of its files. Differently from the SBML case, all the information our system handles are present in the files that can be downloaded from the Reactome website. Unfortunately KEGG does not keep up to date its files. So all the KEGG available BioPax files are dated back 2005, are in the less expressive version 1 of the format and are incomplete with lots of missing pathways. So it would be possible to extract the information of Reactome from the BioPax files but there would be no chance to do a real semantic integration with KEGG.

So we chose the available SOAP based web services. Although the approach does not allow semantic but just syntactical integration, it has anyway lots of good advantages: it is always up to date since the returned data is taken directly from the data sources available on line; the performance of searching data in a huge knowledge base as KEGG are much better using the native API than having to search the information among all the possible XML files whatever way of storing XML documents we use [22]; using the API requires neither any kind of stored files on the machine which hosts the service nor large overnight traffic to keep the local versions of the data sources up to date.

Conclusions

Our system can integrate the information described in its model for the two data sources Reactome and KEGG in a modular, efficient, effective and transparent way.

It is modular because the wrappers can be easily extended to extract more information from the same source thanks to the modeling of the knowledge base in more relations. The whole system can also be extended with a reasonable amount of

work to integrate other data sources modeled as containing information similar to the ones already extracted from KEGG and Reactome.

It is efficient since a fast and reliable GAV rewriting is used and because the wrappers extract information from SOAP web services.

It is effective because the API retrieve more information than the incomplete available XML files. Those information are also up to date contrary to the often dated XML files.

It is transparent since the underlying integration of the sources is completely hidden to the end user, so that our system can be used as a part of a hierarchy of integration web services in order to achieve a wider and wider integration.

We believe that, although the OWL based formats like BioPax will be fundamental in the development of the future integration systems, at the moment ours is the best possible approach to the integration of the two specific data sources presented, KEGG and Reactome.

Methods

The architecture of the system

The architecture of the system can be described following the transformation that a query undergoes. The first step is when the query is created in the front end and passed to the system. The system is exposed as a SOAP based web services implemented with the Java technology but currently the only developed front end is an ad hoc realization for another system [14]. The front end passes the query to the first component of the actual integrator system: a query re-formulator. It analyzes the query and reformulate it in terms of the underlying data sources using a GAV description of them. The third step is the analysis of the reformulated queries and the plan generation. In this phase the query execution engine has to understand how the different parts of the query may be combined. It then optimizes the query selecting the best plan and finally passed the optimized query to the relevant wrappers. The fourth step is the analysis of the optimized query carried out by any of the wrappers which received it. A wrapper is a Java class, and according to the kind of query the wrapper will call a different method to extract the information from the data source. Finally those extracted information are passed back to the result analyzer which merges the retrieved data and pass it back to application which called the service.

The rewriting rules

The main aim of this project was to create a system which provided a common interface to a subset of the very large information available in KEGG and Reactome. In particular there are three main mediated schema set of relations which are available: Pathway, Connection_Molecule and Reaction.

Pathway is a relation which permits to retrieve information about a given pathway. The potential input parameters are: its name, its KEGG and/or Reactome id, the referred Gene Ontology term and, only together with any of these, the species which the pathway is referred to. The output attributes of any retrieved tuples are all the listed input parameters plus the description of the pathways present in Reactome for humans. The rules used to rewrite the query over the mediated schema are:

- ◆ Pathway (PathName, KEGGPathwayID, ReactomePathwayID, Description, Organism, GOTerm) :-
 KEGG1 (PathName, KEGGPathwayID, Organism),
 Reactome1(PathName,ReactomePathwayID,Description, Organism, GOTerm)
- ◆ Pathway_KEGG (PathName, KEGGPathwayID, Organism) :-
 KEGG1 (PathName, KEGGPathwayID, Organism)
- ◆ Pathway_Reac (PathName, ReactomePathwayID, Description, Organism, GOTerm) :-
 Reactome1(PathName,ReactomePathwayID,Description, Organism, GOTerm)

Connection_Molecule is a relation which permits to retrieve all the molecules contained in a certain pathway. This is particularly interesting since it is possible to recursively use this relation to see all the pathways which are connected with a precise one by any common molecule. The potential input parameters are: the KEGG or Reactome pathway id, the name of the molecule in KEGG, and, only together with any of these, the class of the molecule. With the term “class” we refer to one of the four kinds of molecule stored in KEGG: enzyme, gene, compound and glycan. Furthermore it is possible querying the relation using a uniqueID. An “uniqueID” is an identifier of the molecule in another data source which can be used to recognize the same molecule in the two knowledge bases. This parameter can be furnished with or without an additional parameter which specifies which database this external identifier is used by. The output attributes of any retrieved tuples are all the listed input parameters plus the Reactome name of the molecule and its definition and description in KEGG. The rules used to rewrite the query over the mediated schema are:

- ◆ Connection_Molecule (ReactomePathwayID, KEGGPathwayID, MoleculeNameR, MoleculeNameK, UniqueID, Database, Definition, Class, Description) :-
 Reactome3 (ReactomePathwayID, ReactomeMoleculeID , MoleculeNameR, UniqueID, Database),
 KEGG2 (KEGGMoleculeID, MoleculeNameK, UniqueID , Definition, Class, Description),
 KEGG3 (KEGGPathwayID, KEGGMoleculeID, Class)
- ◆ Connection_Molecule_Reac (ReactomePathwayID, ReactomeMoleculeID , MoleculeNameR, UniqueID, Database) :-
 Reactome3 (ReactomePathwayID, ReactomeMoleculeID , MoleculeNameR, UniqueID, Database),
- ◆ Connection_Molecule_KEGG (KEGGPathwayID, KEGGMoleculeID, MoleculeNameK, UniqueID , Definition, Class, Description) :-
 KEGG2 (KEGGMoleculeID, MoleculeNameK, UniqueID , Definition, Class, Description),
 KEGG3 (KEGGPathwayID, KEGGMoleculeID, Class)

Reaction is the name of the last relation. It is a very small relation used to retrieve all the reactions which belong to a determinate pathway. Currently the reactions are retrieved only from Reactome and not from KEGG. The potential input parameters are the pathway name and the pathway id in Reactome. The output attributes of any retrieved tuples are all these attributes plus the reaction name. The rule used to rewrite the query over the mediated schema is:

- ◆ Reaction (PathName, ReactomePathwayID, Reaction) :-

Reactome1(PathName,ReactomePathwayID,Description,Organism, GOTerm),
Reactome2 (ReactomePathwayID, Reaction)

Authors' contributions

EG and LV modeled the mediated schema and drafted the manuscript. LV developed the core application and the wrappers. IP and MF developed the front end of the application. EG, IP and MF supervised the project development. All authors read and approved the final manuscript.

Acknowledgments

We would like to thank Drs. Silvia Scaglione, bioengineering researcher at the Bioengineering and Bioimages laboratory (Biolab) of the University of Genova, and Drs. Federica Viti, PhD student at the Bioengineering and Bioimages laboratory (Biolab) of the University of Genova, actively working at the Institute for Biomedical Technologies of the National Research Council, for their valuable suggestions on the selection of the knowledge bases, the data to be extracted from them and for the indications about their structure.

References

1. Knoblock C.A, Lerman K, Minton S, Muslea I: **Accurately and Reliably Extracting Data from the Web: A Machine Learning Approach.** *Journal of Autonomous Agents and Multi-Agent Systems*, 2001, **4**: 93-114
2. Levy A.Y, Rajaraman A, Ordille J.J: **Query-answering algorithms for information agents.** In: *Proceedings of AAAI-96*: 1996.
3. Knoblock C.A., Minton S, Ambite J.L, Ashish N, Muslea I, Philpot A, Tejada S: **The ariadne approach to web-based information integration.** *International Journal of Cooperative Information Systems*, 2001, **10**: 1–2, 145–169.
4. Strömbäck L, Lambrix P: **Representations of molecular pathways: an evaluation of SBML, PSI MI and BioPAX.** *Bioinformatics*, 2005, **21**: 4401–4407.
5. Doan A, Madhavan J, Domingos P, Halevy A: **Ontology Matching: A Machine Learning Approach.** *Handbook on Ontologies in Information Systems*, Edited by Staab S. and Studer R. Springer-Verlag; 2004. **Invited paper.** 397-416.
6. Etzold T, Ulyanov A, Argos P: **SRS: Information Retrieval System for Molecular Biology Data Banks.** *Methods in Enzymology*, 1996, **266**: 114-128
7. Davidson S.B, Overton G.C, Tannen V, Wong L: **Biokleisli: a digital library for biomedical researchers.** *International Journal on Digital Libraries, Volume 1.* 1997, **1**: 36–53.
8. Kemp G.J.L, Angelopoulos N, Gray P.M.D: **A schema based approach to building a bioinformatics database federation.** *IEEE International Symposium on Bio-Informatics and Biomedical Engineering*, 2000: 13-20.

9. Thakkar S, Ambite J.L, Knoblock C.A: **Composing, optimizing, and executing plans for bioinformatics web services.** *The VLDB Journal*, 2005, **14(3)**: 330–353
10. Stevens R, Goble C, Paton N.W, Bechhofer S, Ng G, Baker P, Brass A: **Complex query formulation over diverse information sources in TAMBIS.** In: *Bioinformatics: Managing Scientific Data*. Edited by Lacroix Z, Critchlow T. San Francisco: Morgan Kaufmann; 2003:189-224.
11. Mork P, Halevy A, Tarczy-Hornoch P: **A model for data integration systems of biomedical data applied to online genetic databases.** In: *Proceedings of the American Medical Informatics Association, Fall Symposium Suppl*, 2001: 473-477.
12. Kanehisa M, Goto S, Kawashima S, Nakaya A: **The KEGG databases at GenomeNet.** *Nucleic Acids Research*, 2002, **30**: 42-46.
13. Joshi-Tope G, Gillespie M, Vastrik I, D'Eustachio P, Schmidt E, De Bono B, Jassal B, Gopinath G.R, Wu G.R, Matthews L, Lewis S, Birney E, Stein L: **Reactome: a knowledgebase of biological pathways.** *Nucleic Acids Research*, 2005, **33**: D428- D432.
14. Beltrame F, Papadimitropoulos A, Porro I, Scaglione S, Schenone A, Torterolo L, Viti F: **GEMMA - A Grid environment for microarray management and analysis in bone marrow stem cells experiments.** *Future Generation Computer Systems, Volume 23*, 2007, **3**: 382-390.
15. Halevy A.Y: **Theory of Answering Queries Using Views.** *SIGMOD Record, Volume 29*, 2000, **4**: 40-47.
16. Garcia-Molina H, Papakonstantinou Y, Quass D, Rajaraman A, Sagiv Y, Ullman J, Vassalos V, Widom J: **The TSIMMIS Approach to Mediation: Data Models and Languages.** *Journal of Intelligent Information Systems*, 1997, **8**: 117–132.
17. Duschka O.M, Genesereth M.R: **Answering recursive queries using views.** In: *Proceedings 16th ACM SIGACT-SIGMOD-SIGART Symposium Principles of Database Systems: May 12-14 1997; Tucson*. 1997: 109-116.
18. Levy A, Mendelzon A.O, Sagiv Y: **Answering queries using views.** In: *Proceedings of the 14th ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems: May 22-25 1995; San Jose*. 1995: 95-104.
19. Abiteboul S, Duschka O: **Complexity of answering queries using materialized views.** In: *Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems: June 1-3, 1998; Seattle*. 1998: 254-263.
20. **Systems Biology Markup Language (SBML)** [<http://sbml.org>]
21. **BioPAX : Biological Pathways Exchange** [<http://www.biopax.org/>]
22. Tian F, De Witt D.J, Chen J, Zhang C: **The Design and Performance Evaluation of Alternative XML Storage Strategies.** *SIGMOD Record*, 2002, **31 (1)**: 5-10.